WHAT IS CLAIMED IS:

1. A processor comprising:

a first pipeline including a first stage at which instruction results are committed to architected state, wherein the first stage is separated from an issue stage of the first pipeline by a first number of stages;

a second pipeline including a second stage at which an exception is reportable, wherein the second stage is separated from the issue stage of the second pipeline by a second number of stages which is greater than the first number; and

a control circuit configured to inhibit co-issuance of a first instruction to the first pipeline and a second instruction to the second pipeline if the first instruction is subsequent to the second instruction in program order.

2. The processor as recited in claim 1 wherein the control circuit is configured to selectively inhibit co-issuance of the first instruction and the second instruction responsive to whether or not exceptions are enabled for the second instruction.

3. The processor as recited in claim 1 wherein the second instruction is a floating point instruction and the second pipeline is a floating point pipeline.

4. The processor as recited in claim 3 wherein the first instruction is an integer instruction and the first pipeline is an integer pipeline.

5. The processor as recited in claim 3 wherein the first instruction is a load/store instruction and the first pipeline is a load/store pipeline.

6. The processor as recited in claim 3 wherein floating point instructions include short floating point instructions having a first latency during execution, a floating point multiply-add instruction having a second latency during execution which is greater than the first latency, and long latency floating point instructions having a third latency during execution which is greater than the second latency, and wherein the second number of stages is dependent on whether the second instruction is one of the short floating point instructions, the floating point multiply-add instruction, or one of the long latency floating point instruction.

7. The processor as recited in claim 6 wherein, if the second instruction is not one of the short floating point instructions, the control circuit is configured to inhibit co-issuance of subsequent floating point instructions, in program order, to a third pipeline.

8. The processor as recited in claim 1 wherein the control circuit is further configured to inhibit subsequent issue of instructions until a predetermined number of clock cycles prior to the second instruction reaching the second stage.

9. The processor as recited in claim 8 further comprising a scoreboard coupled to the control circuit, wherein the control circuit is configured to logically combine the indications in the scoreboard, and wherein the control circuit is configured to permit subsequent issue of instructions responsive to the logical combination having a result indicating that no register writes are pending.

10. The processor as recited in claim 9 wherein the scoreboard comprises a bit for each register indicative, when set, that a write is pending to the register, and wherein the logical combination of the bits is a logical OR, and wherein the control circuit is configured to permit subsequent issue of instructions responsive to the logical OR being zero.

11. A method comprising:

inhibiting co-issuance of a first instruction to a first pipeline and a second instruction to a second pipeline if the first instruction is subsequent to the second instruction in program order;

wherein the first pipeline includes a first stage at which instruction results are committed to architected state, wherein the first stage is separated from an issue stage of the first pipeline by a first number of stages; and wherein the second pipeline includes a second stage at which an exception is reportable, wherein the second stage is separated from the issue stage of the second pipeline by a second number of stages which is greater than the first number.

12. The method as recited in claim 11 wherein the inhibiting is selective responsive to whether or not exceptions are enabled for the second instruction.

13. The method as recited in claim 11 wherein the second instruction is a floating point instruction and the second pipeline is a floating point pipeline.

14. The method as recited in claim 13 wherein the first instruction is an integer instruction and the first pipeline is an integer pipeline.

15. The method as recited in claim 13 wherein the first instruction is a load/store instruction and the first pipeline is a load/store pipeline.

16. The method as recited in claim 13 wherein floating point instructions include short floating point instructions having a first latency during execution, a floating point

multiply-add instruction having a second latency during execution which is greater than the first latency, and long latency floating point instructions having a third latency during execution which is greater than the second latency, and wherein the second number of stages is dependent on whether the second instruction is one of the short floating point instructions, the floating point multiply-add instruction, or one of the long latency floating point instruction, and wherein the method further comprises, if the second instruction is not one of the short floating point instructions, inhibiting co-issuance of subsequent floating point instructions, in program order, to a third pipeline.

17. The method as recited in claim 11 further comprising inhibiting subsequent issue of instructions until a predetermined number of clock cycles prior to the second instruction reaching the second stage.

18. The method as recited in claim 17 further comprising:

logically combining the indications in a scoreboard; and

permitting subsequent issue of instructions responsive to the logical combination having a result indicating that no register writes are pending.

19. The method as recited in claim 18 wherein the scoreboard comprises a bit for each register indicative, when set, that a write is pending to the register, and wherein the logical combining is a logical ORing, and wherein the permitting is responsive to the logical OR being zero.

20. A carrier medium comprising one or more data structures representing a processor, the processor including:

a first pipeline including a first stage at which instruction results are committed to

architected state, wherein the first stage is separated from an issue stage of the first pipeline by a first number of stages;

a second pipeline including a second stage at which an exception is reportable, wherein the second stage is separated from the issue stage of the second pipeline by a second number of stages which is greater than the first number; and

a control circuit configured to inhibit co-issuance of a first instruction to the first pipeline and a second instruction to the second pipeline if the first instruction is subsequent to the second instruction in program order